

Politechnika Świętokrzyska w Kielcach

Wydział Elektrotechniki, Automatyki i Informatyki

Modelowanie i analiza systemów informatycznych 2025

Temat: Modelowanie i analiza systemu informatycznego realizującego zamianę unitermu poziomej operacji eliminowania unitermów na poziomą operację zrównoleglania unitermów

Autor:

Oskar Andrzejewski

Grupa:

1ID24B

Spis treści

1. Wstęp.....	3
2. Opis kodu.....	4
3. Diagramy.....	12
3.1. Diagram przypadków użycia.....	12
3.2. Diagram klas.....	13
3.3. Diagram aktywności.....	14
3.4. Diagram sekwencji.....	15
3.5. Diagram warstw.....	16
3.6. Diagram komponentów.....	18
4. Baza danych.....	19
5. Zdjęcia z działania aplikacji.....	20
6. Wnioski.....	23

1. Wstęp

Unitermy stanowią uporządkowane wyrażenia logiczne lub językowe, które mogą być modyfikowane za pomocą specyficznych operacji. Dwie z podstawowych operacji to eliminacja oraz zrównoleglenie.

Celem projektu było stworzenie interaktywnego narzędzia umożliwiającego wizualizację oraz przekształcanie unitermów poprzez zamianę poziomej operacji eliminowania na odpowiadającą jej operację zrównoleglania. W tym celu opracowano aplikację w języku Python z wykorzystaniem biblioteki Tkinter, która pozwala użytkownikowi na:

- Wprowadzenie tekstów reprezentujących poszczególne części unitermów.
- Graficzne przedstawienie operacji eliminacji i zrównoleglania.
- Dokonanie transformacji jednej z części unitermu na jej zrównoleglenie za pomocą interfejsu graficznego.
- Zapisanie wyniku operacji oraz danych wejściowych do zewnętrznej bazy danych Supabase.

2. Opis kodu

Biblioteki:

- tkinter – biblioteka wykorzystana w celu stworzenia GUI
- supabase – biblioteka umożliwiająca korzystanie z API chmurowe bazy danych Supabase

Klasa MASI_PRO – Główna klasa projektu, odpowiada za przechowywanie danych, obsługę GUI oraz komunikację z pozostałymi klasami pomocniczymi.

Metody Klasy:

- `create_menu(self)` – Dodaje do okno pasek menu
- `clear_canvas(self)` – Usuwa wygenerowane linie oraz usuwa zapisane unitermy
- `open_modal(self, mode)` – Otwiera odpowiednie okno modalne w celu dodania unitermów operacji eliminacji lub zrównoleglenia wykorzystując klasę `ModalWindow`, rodzaj operacji jest ustalany na podstawie przekazanego argumentu *mode*
- `handle_modal_response(self, mode, text, parts, separator)` – Generuje graficznie unitermy oraz operacje na podstawie danych wprowadzonych przez użytkownika. Argument *mode* odpowiada za rodzaj operacji, *parts* zawiera wpisane przez użytkownika unitermy, *separator* odpowiada za znak rozdzielający unitermy
- `handle_dialog(self, option)` – Obsługuje odpowiedź z okno modalnego klasy wygenerowanego w klasie `Menu_Modal`, zamienia wybrany przez użytkownika uniterm operacji eliminacji na operację zrównoleglenia i wykorzystuje klasę `DatabaseHandler` do przesłania informacji do bazy danych. Argument *option* zawiera informacje, który uniterm operacji eliminacji ma być zastąpiony
- `draw_uniterm(self, copy_line=False, shift_line=False)` – Generuje tekst oraz linie na ekranie aplikacji, obsługuje tworzenie zarówno unitermów operacji zrównoleglenia, eliminacji przed i po podstawieniu. Wartości *copy_line* i *shift_line* pozwalają na kolejno odpowiednią generację linii przy zamianie unitermu eliminacji oraz modyfikację tekstu eliminacji w przypadku podstawienia
- `update_font(self, *_)` - Zbiera dane wprowadzone przez użytkownika z okien wyboru czcionki i jej rozmiaru oraz ponownie generuje operacje z wykorzystaniem nowej czcionki

```

class MASI_PRO:
    def __init__(self, root):
        self.root = root
        self.root.title("MASI_PRO_17_OA")

        main_area = tk.Frame(root)
        main_area.pack(side=tk.LEFT, fill=tk.BOTH, expand=True)

        self.canvas = tk.Canvas(main_area, width=400, height=400)
        self.canvas.pack(side=tk.TOP, fill=tk.BOTH, expand=True)

        self.text_elimiacja = ""
        self.text_zrownoleglenia = ""

        self.separator_elimiacja = ""
        self.separator_zrownoleglenia = ""

        self.parts_elimiacja = []
        self.original_parts_elimiacja = []
        self.parts_zrownoleglenia = []

        self.current_font_family = "Arial"
        self.current_font_size = 20

        self.font_frame = tk.Frame(root)
        self.font_frame.pack(side=tk.RIGHT, fill=tk.Y, padx=10, pady=10)

        Label(self.font_frame, text="Rodzaj czcionki:").pack(pady=5)
        font_families = list(font.families())
        self.font_family_var = StringVar()
        self.font_family_var.set(self.current_font_family)
        font_menu = OptionMenu(self.font_frame, self.font_family_var, *sorted(font_families),
command=self.update_font)
        font_menu.pack()

        Label(self.font_frame, text="Rozmiar czcionki:").pack(pady=5)
        self.font_size_var = tk.IntVar()
        self.font_size_var.set(self.current_font_size)
        font_size_spinbox = Spinbox(self.font_frame, from_=8, to=72,
textvariable=self.font_size_var, command=self.update_font)
        font_size_spinbox.pack()

        self.modal = Menu_Modal(self.root, self)
        self.create_menu()

        self.button_frame = tk.Frame(main_area)
        self.button_frame.pack(side=tk.BOTTOM, fill=tk.X, pady=10)

        center_frame = tk.Frame(self.button_frame)
        center_frame.pack()

```

```

        self.eliminacja_button = tk.Button(center_frame, text="Eliminacja", command=lambda:
self.open_modal("eliminacja"))
        self.eliminacja_button.pack(side=tk.LEFT, padx=10)

        self.zrownoleglenia_button = tk.Button(center_frame, text="Zrownoleglenie",
command=lambda: self.open_modal("zrownoleglenia"))
        self.zrownoleglenia_button.pack(side=tk.LEFT, padx=10)

        self.clear_button = tk.Button(center_frame, text="Wyczyść", command=self.clear_canvas)
        self.clear_button.pack(side=tk.LEFT, padx=10)

        self.db_handler = DatabaseHandler()
        self.modal_win = ""

def create_menu(self):
    menubar = Menu(self.root)
    file_menu = Menu(menubar, tearoff=0)
    file_menu.add_command(label="Połącz", command=self.modal.open_confirmation_dialog)
    menubar.add_cascade(label="File", menu=file_menu)
    self.root.config(menu=menubar)

def clear_canvas(self):
    self.canvas.delete("all")
    self.text_elimincja = ""
    self.text_zrownoleglenia = ""
    self.parts_elimincja = []
    self.original_parts_elimincja = []
    self.parts_zrownoleglenia = []

def open_modal(self, mode):
    self.modal_win = ModalWindow(self.root, mode, self.handle_modal_response)

def handle_modal_response(self, mode, parts, separator):
    text = separator.join(parts)
    if mode == "eliminacja":
        self.text_elimincja = text
        self.parts_elimincja = parts[:]
        self.original_parts_elimincja = parts[:]
        self.separator_elimincja = separator
    else:
        self.text_zrownoleglenia = text
        self.parts_zrownoleglenia = parts
        self.separator_zrownoleglenia = separator
    self.draw_uniterm()

def handle_dialog(self, option):
    odp = option

```

```

if option:
    if self.parts_eliminacja and self.parts_zrownoleglenia:
        self.parts_eliminacja = self.original_parts_eliminacja[:]
        self.parts_eliminacja[0] = self.text_zrownoleglenia
        self.text_eliminacja = self.separator_eliminacja.join(self.parts_eliminacja)
        self.draw_uniterm(copy_line=True)
    else:
        if len(self.parts_eliminacja) > 1 and self.parts_zrownoleglenia:
            self.parts_eliminacja = self.original_parts_eliminacja[:]
            self.parts_eliminacja[1] = self.text_zrownoleglenia
            self.text_eliminacja = self.separator_eliminacja.join(self.parts_eliminacja)
            self.draw_uniterm(copy_line=True, shift_line=True)
            odp = False

text_teliminacja_ori = self.separator_eliminacja.join(self.original_parts_eliminacja)
data = {
    'created_at': 'now()',
    'eliminacja': text_teliminacja_ori,
    'zrownoleglenie': self.text_zrownoleglenia,
    'polaczenie': odp,
    'wynik': self.text_eliminacja,
}

self.db_handler.insert_operation(data)

def draw_uniterm(self, copy_line=False, shift_line=False):
    self.canvas.delete("all")
    x_start = 50
    y_text = 80
    shift_amount = 0
    separator_width = 10

    selected_font = (self.current_font_family, self.current_font_size, "bold")

    if self.text_eliminacja:
        text_id = self.canvas.create_text(x_start, y_text, text=self.text_eliminacja, anchor="w",
font=selected_font)
        bbox = self.canvas.bbox(text_id)
        if bbox:
            text_width = bbox[2] - bbox[0]
            y_line = bbox[1] - 13
            self.canvas.create_line(x_start - 5, y_line, x_start + text_width + 5, y_line, width=4)
            self.canvas.create_line(x_start - 5, y_line - 5, x_start - 5, y_line + 5, width=4)
            self.canvas.create_line(x_start + text_width + 5, y_line - 5, x_start + text_width + 5,
y_line + 5, width=4)

            if shift_line and self.original_parts_eliminacja:
                text_id = self.canvas.create_text(x_start, y_text,
text=self.original_parts_eliminacja[0], anchor="w", font=selected_font)
                bbox = self.canvas.bbox(text_id)

```

```

        if bbox:
            shift_amount = bbox[2] - bbox[0] + separator_width

    if self.text_zrownoleglenia:
        y_text = 130
        text_id = self.canvas.create_text(x_start, y_text, text=self.text_zrownoleglenia,
        anchor="w", font=selected_font)
        bbox = self.canvas.bbox(text_id)
        if bbox:
            text_width = bbox[2] - bbox[0]
            y_line = bbox[1] - 10
            self.canvas.create_line(x_start - 5, y_line, x_start + text_width + 5, y_line, width=4)
            self.canvas.create_line(x_start - 5, y_line - 2, x_start - 5, y_line + 15, width=4)
            self.canvas.create_line(x_start + text_width + 5, y_line - 2, x_start + text_width + 5,
            y_line + 15, width=4)

        if copy_line:
            self.canvas.create_line(x_start - 2 + shift_amount, 57, x_start + text_width + 2 +
            shift_amount, 57, width=4)
            self.canvas.create_line(x_start - 2 + shift_amount, 55, x_start - 2 + shift_amount, 72,
            width=4)
            self.canvas.create_line(x_start + text_width + 2 + shift_amount, 55, x_start +
            text_width + 2 + shift_amount, 72, width=4)

    def update_font(self, *_):
        self.current_font_family = self.font_family_var.get()
        self.current_font_size = self.font_size_var.get()
        self.draw_uniterm()

```


Klasa ModalWindow – Klasa obsługująca okno modalne, w którym użytkownik podaje dane unitermów

Metody Klasy:

- `create_modal(self)` – Tworzy okno modalne, rozróżniając czy użytkownik chce dodać operację eliminacji, czy zrównoleglenia, w oknie użytkownik wprowadza informacje o tworzonym unitermie
- `on_ok(self, modal)` – Obsługuje zapis danych wprowadzonych przez użytkownika oraz zamknięcie okna modalnego przy wciśnięci przyciski ok. Argument *modal* pozawala na zamknięcie okna modalnego po zaakceptowaniu danych

```
class ModalWindow:
    def __init__(self, root, mode, callback):
        self.root = root
        self.mode = mode
        self.callback = callback
        self.entries = []
        self.separator_var = StringVar()
        self.separator_var.set(";")
        self.create_modal()

    def create_modal(self):
        modal = Toplevel(self.root)
        modal.title(self.mode.capitalize())

        num_entries = 3 if self.mode == "eliminacja" else 2
        for i in range(num_entries):
            Label(modal, text=f"Tekst {i + 1} :").grid(row=i, column=0)
            entry = Entry(modal)
            entry.grid(row=i, column=1)
            self.entries.append(entry)

        Label(modal, text="Separator:").grid(row=num_entries, column=0)
        separator_menu = OptionMenu(modal, self.separator_var, ";", ",")
        separator_menu.grid(row=num_entries, column=1)

        Button(modal, text="OK", command=lambda: self.on_ok(modal)).grid(row=num_entries + 1,
            columnspan=2, pady=10)

    def on_ok(self, modal):
        parts = [entry.get() for entry in self.entries if entry.get()]
        separator = self.separator_var.get()
        self.callback(self.mode, parts, separator)
        modal.destroy()
```

Klasa Menu_Modal – Klasa obsługująca okno modalne, w którym użytkownik wybiera pod który uniterm operacji eliminacji, podstawiona operacja zrównoleglenia unitermów.

Metody Klasy:

- `open_confirmation_dialog(self)` – Generuje okno modalne pozwalające na wybranie przez użytkownika który uniterm operacji eliminacji ma być zastąpiony operacją zrównoleglenia, po wyborze przycisku okno automatycznie jest zamykane.
- `uni_1()` - Obsługa 1 przycisku, wywołująca `handle_dialog(True)` w głównym oknie w celu podstawienia operacji zrównoleglenia za 1 uniterm operacji eliminacji
- `uni_2()` - Obsługa 2 przycisku, wywołująca `handle_dialog(False)` w głównym oknie w celu podstawienia operacji zrównoleglenia za 2 uniterm operacji eliminacji
- `on_cancel()` - Obsługa 3 przycisku odpowiadającego za anulowanie wyboru

```
class Menu_Modal:
    def __init__(self, root, main_window):
        self.root = root
        self.main_window = main_window

    def open_confirmation_dialog(self):
        modal = Toplevel(self.root)
        modal.title("Podstaw")
        modal.geometry("400x100")
        Label(modal, text="Za który uniterm chcesz podstawić operację
zrównoleglenia?").pack(pady=10)

        def uni_1():
            self.main_window.handle_dialog(True)
            modal.destroy()

        def uni_2():
            self.main_window.handle_dialog(False)
            modal.destroy()

        def on_cancel():
            print("Anulowano")
            modal.destroy()

        button_frame = tk.Frame(modal)
        button_frame.pack(pady=10)

        tk.Button(button_frame, text="1 uniterm", command=uni_1).pack(side=tk.LEFT, padx=5)
        tk.Button(button_frame, text="2 uniterm", command=uni_2).pack(side=tk.LEFT, padx=5)
        tk.Button(button_frame, text="Anuluj", command=on_cancel).pack(side=tk.LEFT, padx=5)
```

Klasa DatabaseHandler – przechowuje url oraz klucz potrzebne do dostępu do bazy danych oraz obsługują połączenie z bazą Supabase.

Metody Klasy:

- `insert_operation(self, data: dict)` – Przesyła dane dotyczące operacji podstawienia do chmurowej bazy danych Supabase i zapisuje je w tabeli Operacje. Argument *data* zawiera dane przesyłane do bazy danych

```
class DatabaseHandler:
    def __init__(self):
        self.supabase_url = "https://lxrnycgqiejbhjfbdxcz.supabase.co"
        self.supabase_key =
("eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJkdXBhYmFzZSI6InJlZiI6Imx4cm55Y2dxZWVqYmhhZmJkeGN6Iiwicm9sZSI6InNlcnZpY2VfcmlhdCI6MTc0Mjc5OTQzNywiZXBhYmFzZSI6MDU4Mzc1NDM3fQ.pXO3pWzZeU8VOK7RPXNozduuWz2lAdm4-l0fgk2PIuM")
        self.client: Client = create_client(self.supabase_url, self.supabase_key)

    def insert_operation(self, data: dict):
        print("Dane przesyłane:", data)
        result = self.client.table('Operacje').insert(data).execute()
        return result
```

3. Diagramy

3.1. Diagram przypadków użycia

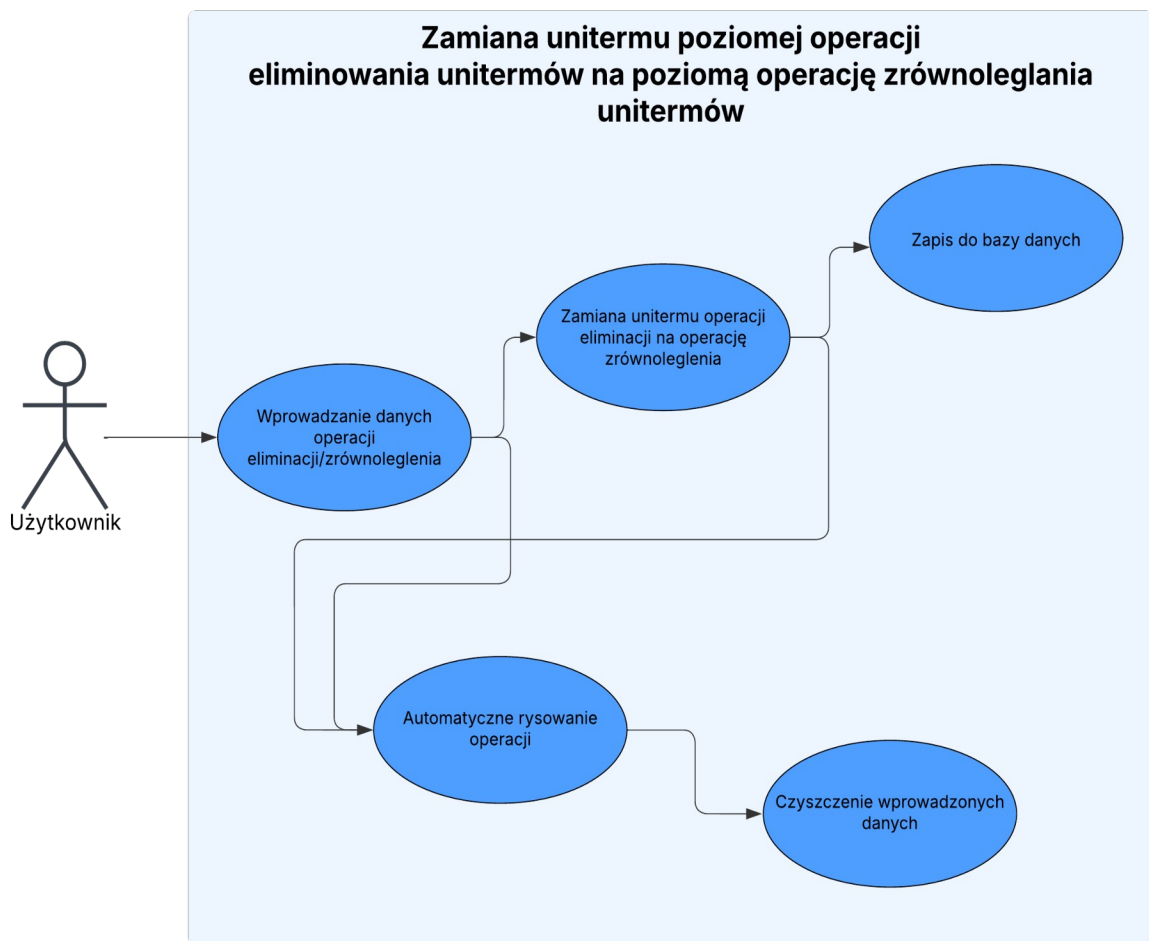


Diagram przedstawia proces zamiany unitermu poziomej operacji eliminacji na poziomą operację zrównoleglenia unitermów. Użytkownik inicjuje proces poprzez wprowadzenie danych dotyczących operacji eliminacji oraz operacji zrównoleglenia. Następnie aplikacja umożliwia zamianę dowolnego unitermu operacji eliminacji, automatycznie wysyła informacje o operacji do bazy danych oraz automatycznie rysuje operację po każdej zmianie wprowadzonej przez użytkownika, pozwalając również na czyszczenie danych w dowolnym momencie.

3.2. Diagram klas

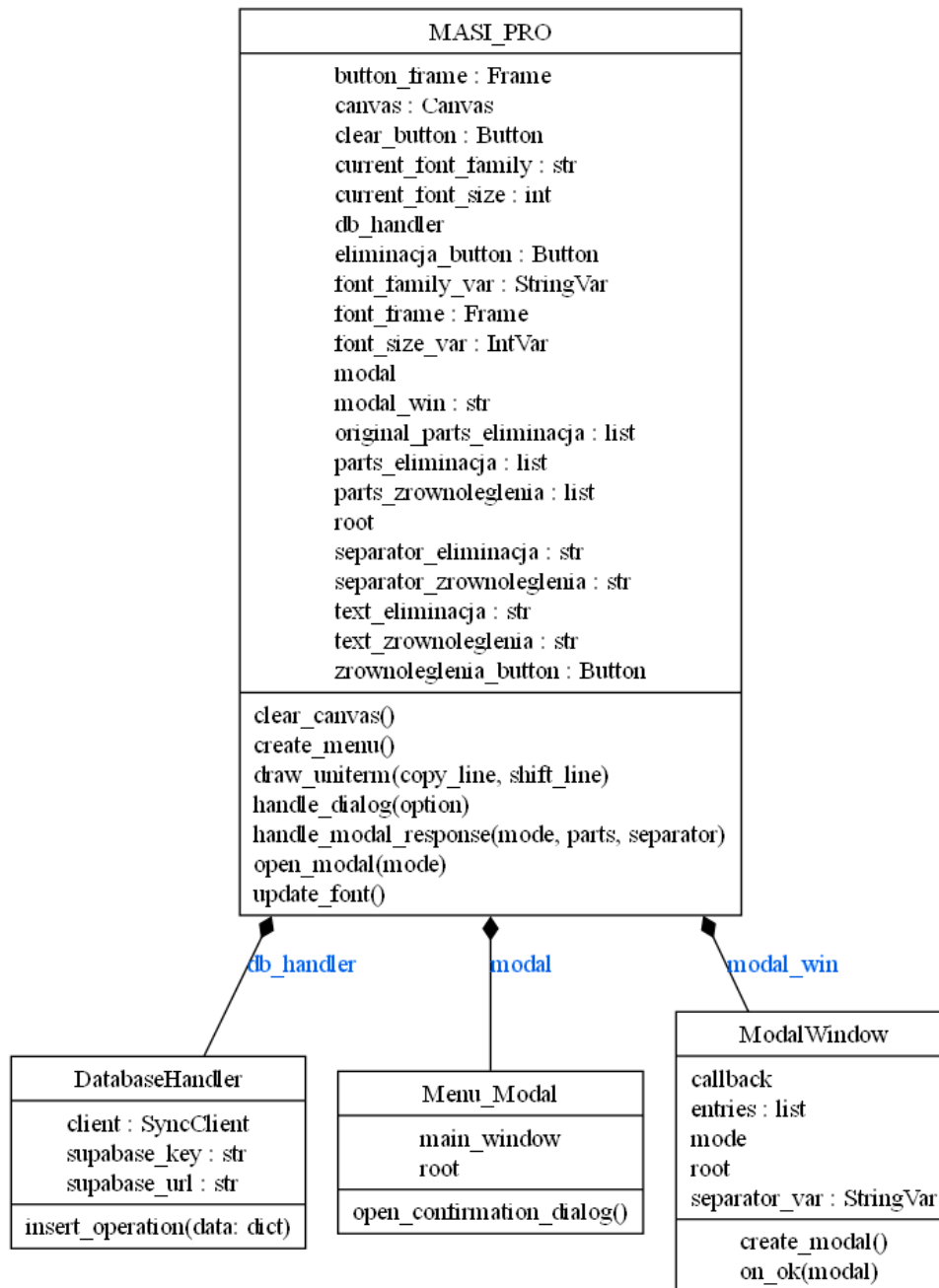


Diagram przedstawia wykorzystane w programie klasy wraz z ich metodami.

Przedstawiona jest również kompozycja klas:

- MASI_PRO komponuje:
 - DatabaseHandler jako db_handler – odpowiada za operacje na bazie danych
 - Menu_Modal jako modal – odpowiada za dialog wyboru unitermu do zastąpienia
 - ModalWindow jako modal_win – reprezentuje okno do wprowadzania danych przez użytkownika

3.3. Diagram aktywności

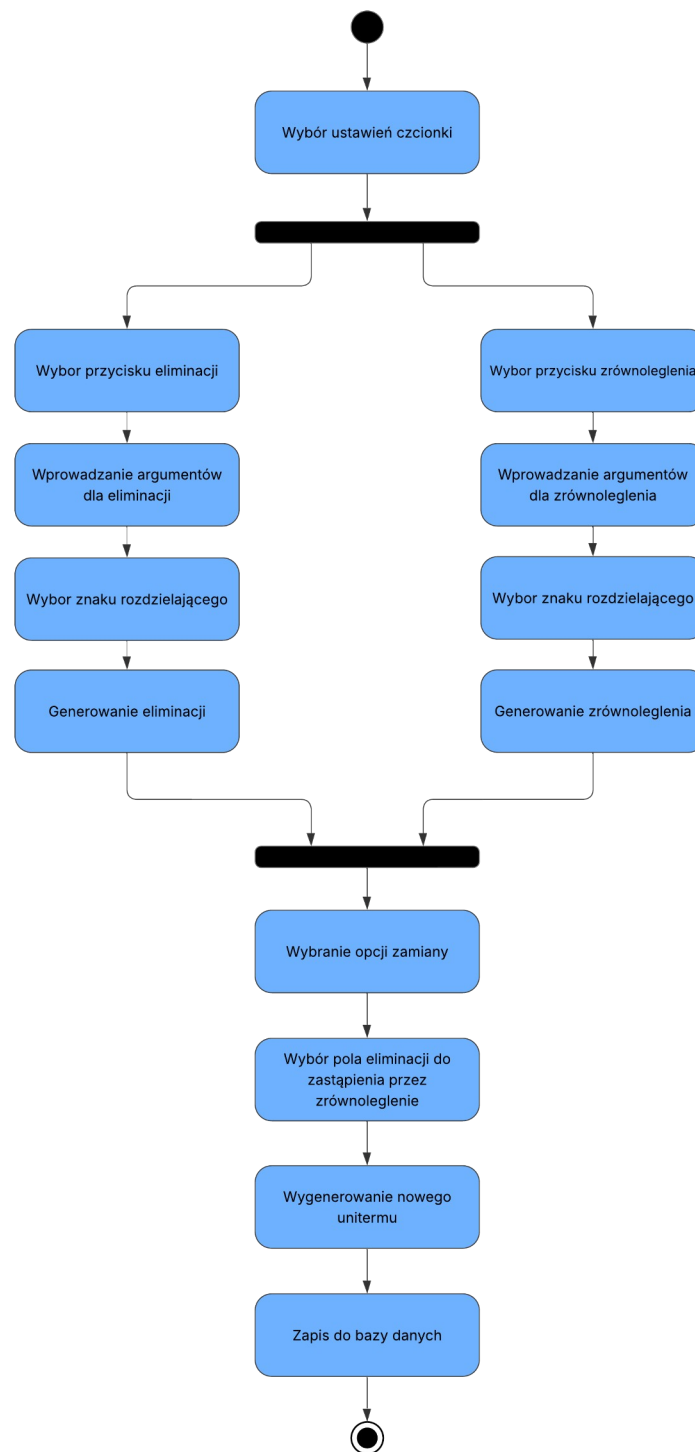


Diagram przedstawia kolejne etapy działania programu od jego włączenia do utworzenia finalnej zamiany unitermu operacji eliminacji na operację zrównoleglenia i zapisaniu działań w chmurowej bazie danych. Kolejnymi etapami są: wybór ustawień czcionki, podanie unitermów operacji eliminacji i zrównoleglenia, narysowanie tych operacji, wybranie przez użytkownika który uniterm eliminacji ma być zastąpiony oraz wysłanie informacji o operacji do bazy danych.

3.4. Diagram sekwencji

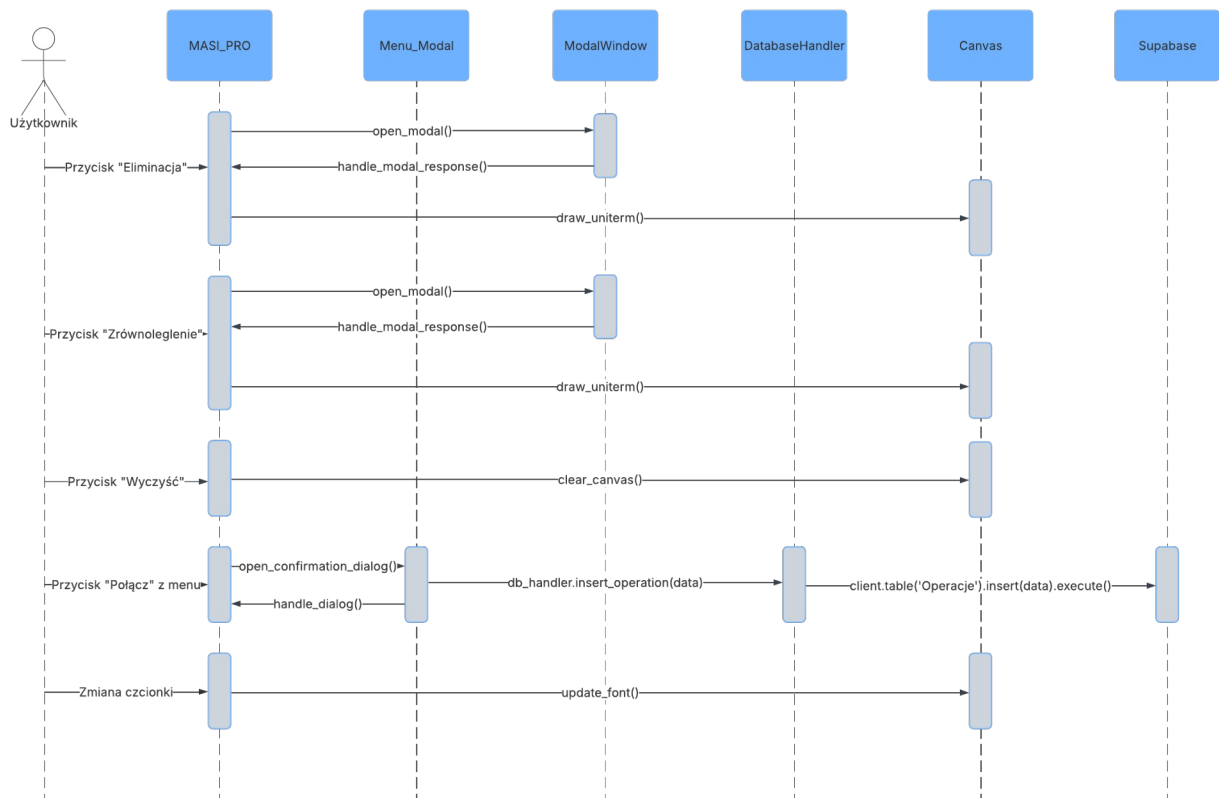


Diagram sekwencji przedstawia proces zamiany unitermu poziomej operacji eliminowania unitermów na poziomą operację zrównoleglania unitermów. W interakcję wchodzi użytkownik, klasy programu MASI_PRO, ModalWindow, Menu_Modal, DatabaseHandler oraz chmurowa baza danych Supabase. Użytkownik korzysta z aplikacji poprzez wykorzystanie przycisków, które pozwalają na zmianę czcionki, czyszczenie ekranu, włączanie okien modalnych klas Menu_Modal oraz ModalWindow które pozwalają na podanie danych do stworzenia operacji eliminowania i zrównoleglania unitermów. Menu_Modal po wybraniu podstawienia automatycznie wykorzystuje DatabaseHandler w celu przesłania danych o operacji do bazy danych Supabase.

3.5. Diagram warstw

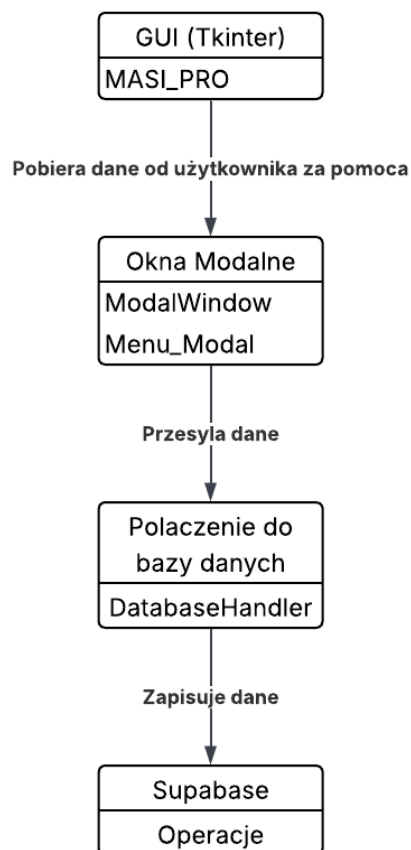


Na program składają się 4 główne warstwy:

- Warstwa GUI – Odpowiada za prezentację i interakcję z użytkownikiem. Zrealizowana przy użyciu biblioteki tkinter, obejmuje klasy i metody odpowiedzialne za tworzenie i konfigurację okien modalnych, przycisków, pól tekstowych, menu oraz płótna (Canvas) do wizualizacji danych obsługę wyboru czcionki i jej rozmiaru
- Warstwa logiki aplikacji – Zawiera metody przetwarzające dane wprowadzone przez użytkownika obsługuje: przetwarzanie tekstów eliminacji i zrównoleglenia, ich łączenie w zależności od wybranej opcji przygotowanie danych do wizualizacji oraz logikę rysowania na płótnie

- Warstwa dostępu do bazy danych – Realizowana poprzez klasę DatabaseHandler. Zapewnia interfejs do komunikacji z bazą danych poprzez klienta Supabase, wysyła ona dane do bazy danych za pomocą operacji insert
- Warstwa Bazy Danych – Reprezentuje zewnętrzne źródło danych – chmurową bazę Supabase. W tej warstwie przechowywane są historyczne operacje wykonane w programie









3.6. Diagram komponentów



Aplikacja składa się z 4 głównych komponentów:

- GUI – prezentacja aplikacji, rysowanie danych na płótnie oraz obsługę zdarzeń użytkownika. Klucowym elementem jest klasa **MASI_PRO** która zawiera przyciski, menu oraz rysowanie operacji unitermów
- Okna Modalne – umożliwienie użytkownikowi wprowadzenia danych (unitermów i separatorów) potrzebnych do wykonania operacji logicznych. Kluczowymi elementami są klasy odpowiadające za okna modalne: **ModalWindow** oraz **Menu_Modal**
- Połączanie do bazy danych – umożliwia wysyłanie danych o operacji do chmurowej bazy danych. Klucowym elementem jest klasa **DatabaseHandler** obsługująca to połączenie
- Supabase – Supabase to backend-as-a-service bazujący na PostgreSQL, używany jako zewnętrzna baza danych w projekcie. Klucowym elementem jest tabela **Operacje** przechowująca zapisane operacje

4. Baza danych

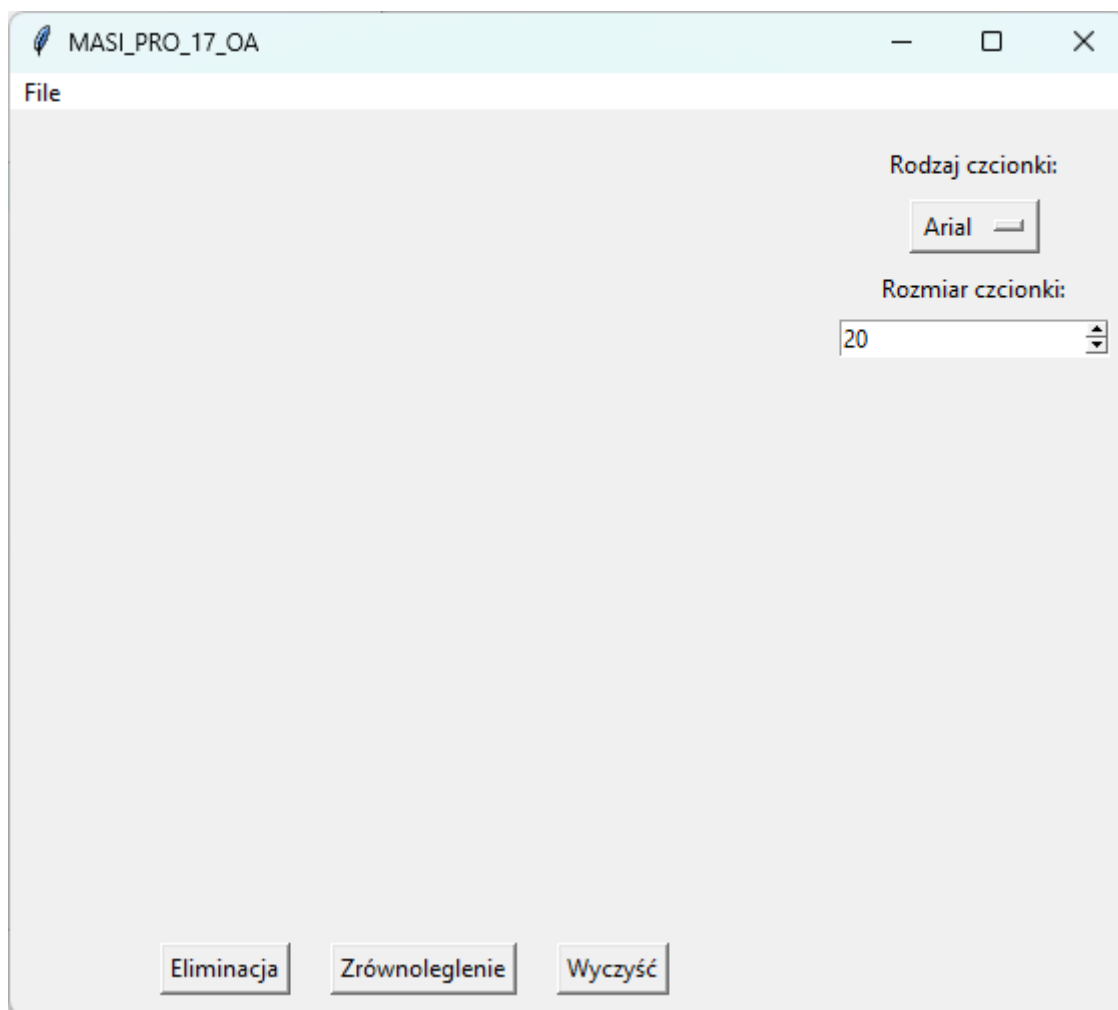
Operacje		
  #	id	ints
	created_at	timestamptz
	eliminacja	text
	zrownoleglenie	text
	polaczenie	bool
	wynik	text

Jako chmurową bazę danych wykorzystana została platforma Supabase. Wykorzystanie rozwiązania chmurowego pozwala na zdalny dostęp do danych oraz odciąża to system poprzez przechowywanie danych na chmurze.

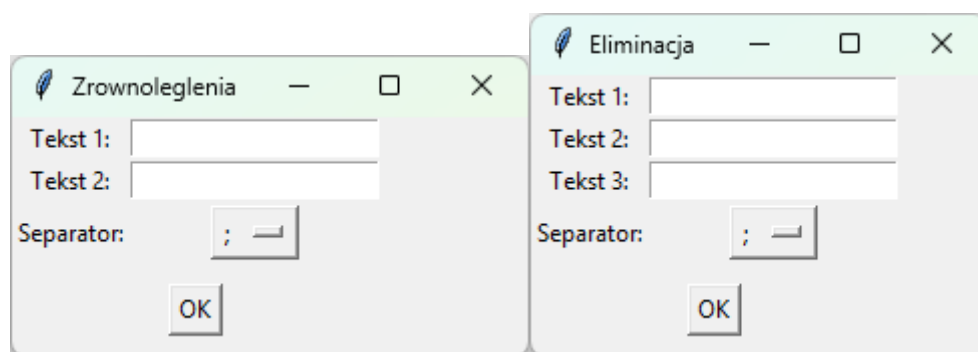
Baza danych składa się z tabeli „Operacje”. Zawiera ona pola:

- id – identyfikator operacji
- created_at – data utworzenia wpisu
- eliminacja – unitermy tworzące operacje eliminacji
- zrownoleglenie – unitermy tworzące operację zrównoleglenia
- polaczenie – wartość boolean zapisująca który uniterm operacji eliminacji został zastąpiony
- wynik – unitermy tworzące operację eliminacji po podstawieniu

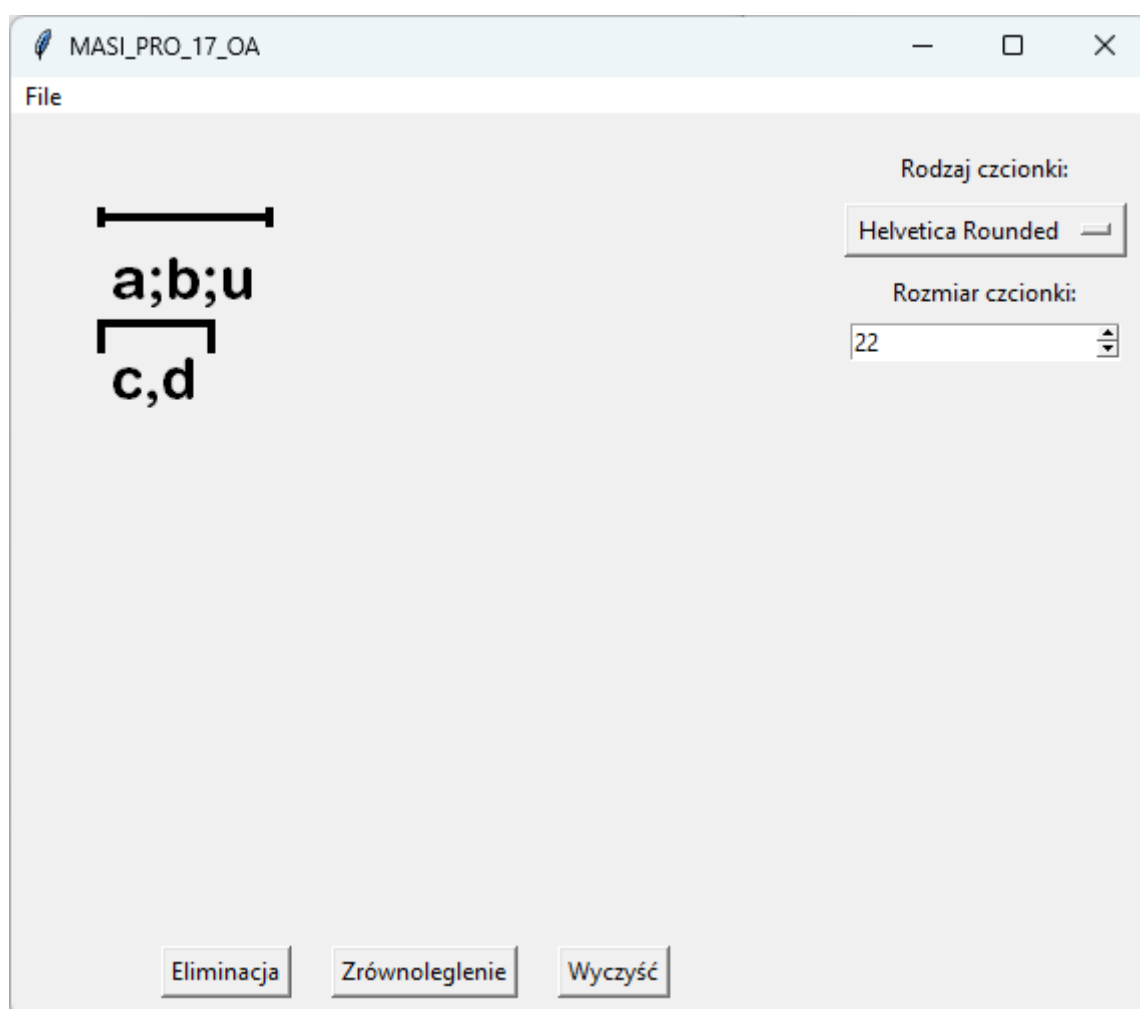
5. Zdjęcia z działania aplikacji



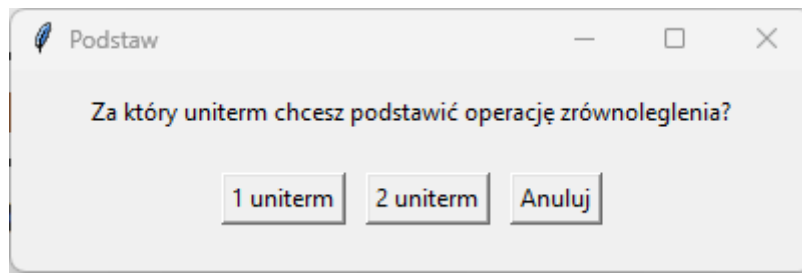
Główne okno aplikacji bez wpisanych danych



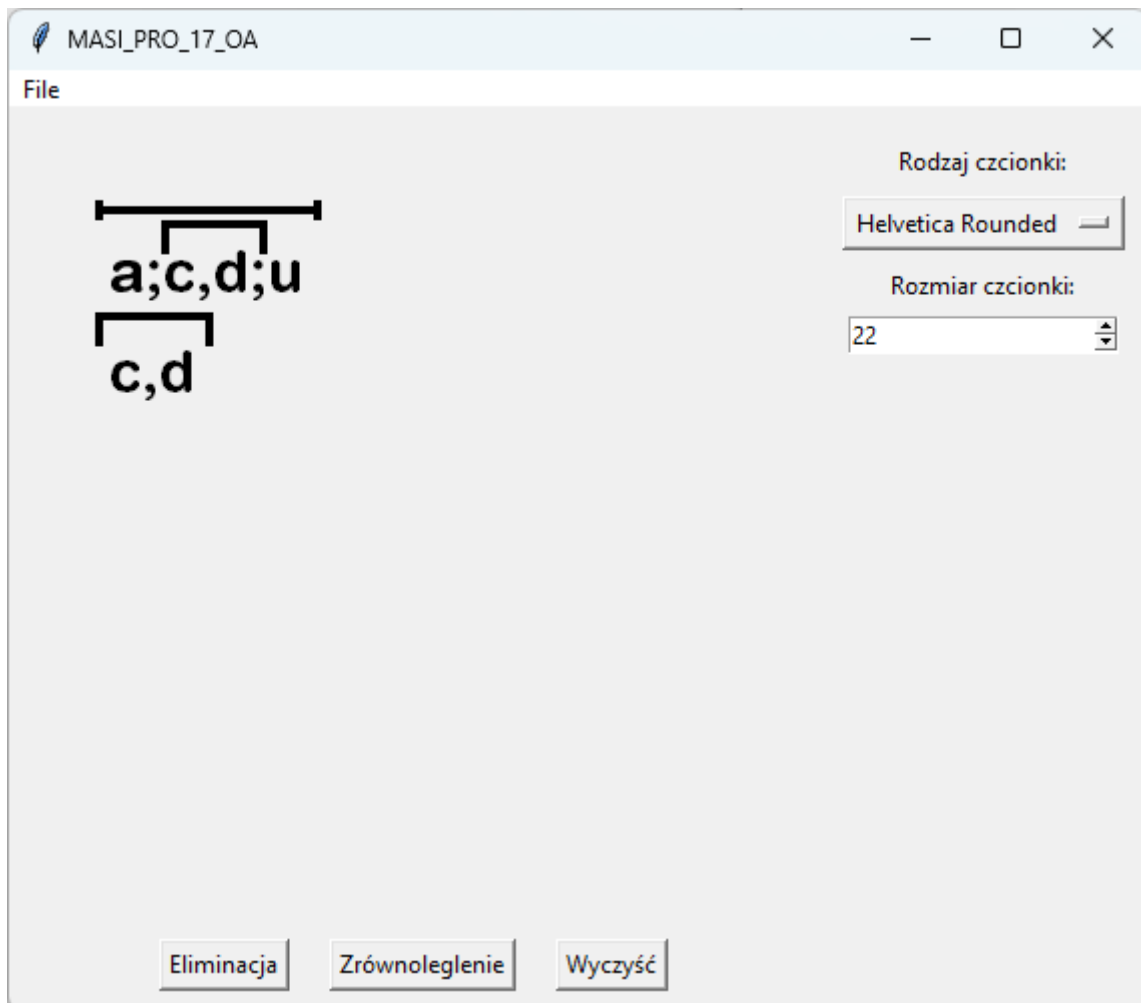
Okna modalne pozwalające na wprowadzanie unitermów operacji Zrównoleglenia i Eliminacji



Unitermy przed połączeniem



Okno modalne wyboru zamiany unitermu eliminacji na operację zrównoleglenia



Przykładowa zamiana unitermu operacji eliminacji na operację zrównoleglenia

6. Wnioski

Zrealizowano model systemu informatycznego, który dokonuje zamiany unitermu poziomej operacji eliminowania unitermów na poziomą operację zrównoleglania unitermów. Stworzony system został przeanalizowany i na jego podstawie zostały stworzone diagramy: przypadków użycia, klas, aktywności, sekwencji, warstw oraz komponentów. Pozwoliły one na prostsze projektowanie aplikacji oraz na opis jej działania w prosty i czytelny sposób.